MRoCS: a new Multi-Robot Communication System based on Passive Action Recognition

Barnali Das[a,*], Micael S. Couceiro[b,c], Patricia A. Vargas[a]

[a]*Robotics Lab, School of Mathematical and Computer Sciences, Heriot-Watt University, Edinburgh, UK*
[b]*Institute of Systems and Robotics (ISR), University of Coimbra, Portugal*
[c]*Ingeniarius, Ltd., Mealhada, Portugal*

Abstract

Multi-robot search-and-rescue missions often face major challenges in adverse environments due to the limitations of traditional implicit and explicit communication. This paper proposes a novel multi-robot communication system (MRoCS), which uses a passive action recognition technique that overcomes the shortcomings of traditional models. The proposed MRoCS relies on individual motion, by mimicking the waggle dance of honey bees and thus forming and recognising different patterns accordingly. The system was successfully designed and implemented in simulation and with real robots. Experimental results show that, the pattern recognition process successfully reported high sensitivity with good precision in all cases for three different patterns thus corroborating our hypothesis.

Keywords: Swarm Robotics, Multi-robot communication, Passive action recognition, Bio-inspired computation, Honey bee waggle dance, Unmanned aerial vehicle.

1. Introduction

Search-and-rescue (SaR) missions using a multi-robot-system (MRS) are considered to be very challenging as communication amongst robots is limited due to the fact that the environment is often prone to sensory noise with limited information. Examples of SaR scenarios include earthquakes, floods or other natural disasters and multi-robot-systems could help in the task of rescuing people by aiding fire brigades, ambulances, police forces, and volunteers [43].

A MRS consists of a number of intelligent, self-organised and collaborative robots. Multiple robots can perform complex tasks with a minimum time span and can increase their robustness in the environment [9]. In MRS, individual robots are smart enough to make decisions and can plan to accomplish a complex task collectively. They rely on mutual interactions, as well as the local information from the environment, where the loss of a given robot does not affect the overall system.

*Corresponding author
Email addresses: dasbhowmik.barnali@gmail.com (Barnali Das), micaelcouceiro@isr.uc.pt (Micael S. Couceiro), p.a.vargas@hw.ac.uk (Patricia A. Vargas)

In a swarm, robots dynamically assign themselves to different tasks to fulfil the requirements in a particular environment and conditions [13]. Nevertheless, with the increase of number of robots, as it typically occurs in a swarm of robots, coordination and communication are necessary to fulfil complex tasks. Communication has a huge impact on the performance of a swarm, where the robots interact with each other to exchange knowledge about their environment [37]. However, in such dynamically changing environment, traditional robot communication often reaches its bottleneck and there may be uncertainty on account of incorrect information or information not reaching the robots within the swarm.

Robot communication can be divided into three categories, *(1)* explicit communication (robots directly and intentionally communicate the relevant information to their teammates through some active means)[1, 24, 51, 36], *(2)* implicit communication (robots sense the effects of their teammates' actions through the influence they leave on the environment *e.g.,* stigmergy)[12, 35], and *(3)* passive action recognition (robots use sensors to directly observe the actions of their teammates).

Explicit communication has been the one widely used due to its directness and ease with which robots become aware of the actions and/or goals of their teammates [10]. However, explicit communication shows limitations in terms of fault-tolerance and reliability, as it typically depends upon a noisy, limited-bandwidth communication channel that may be unable to continually maintain all members of the robot swarm connected.

On the other hand, implicit communication is non-transient and needs no encoding or decoding, knowledge of place, or memory. Robots only react to the local configuration of the environment [4]. However, with the increase in number of robots, the interactions also increase, which decrease the response time. As a result, robots are unable to achieve given tasks [27]. Additionally, the robots, which have no knowledge to detect if whether or not the task has been completed, may jeopardise the objectives of the entire swarm.

For instance, let us consider a military applications, wherein robots need to exchange messages. An explicit communication is vulnerable as it can be intercepted, or understood, by opposing forces. From a security perspective, any open implicit or explicit communication method can be jammed, intercepted or otherwise disturbed relatively easily by the enemy. The security of wireless communication has been well researched, but the security of unconventional and more exotic interaction methods should be explored and presents a compelling security challenge [21]. Therefore, other types of communication, such as passive action recognition, is useful and not easily interpretable. Shim and Arkin [46] advocated that a biologically inspired behaviour as a robotic deception system for military application can lead to a robust passive action recognition based communication, which can be beneficial to improve the security system.

Passive action recognition techniques do not rely on any communication medium, language or environmental configuration [22]. However, for such communications to be successful, robots need to be able to recognise teammates' behaviours by decoding and interpreting their actions [34]. Hence, in addition to military applications, this is also useful for hazardous environment, such as, in case of a natural disaster, where establishing communication channel is challenging and sometimes impossible. Human-robot interaction can also be made easy by passive communication, especially when humans are not connected to any other

traditional robot communication medium. This type of swarm behaviour can be observed in nature *e.g.,* honey bee's waggle dance [53, 14] which was used as a bio-inspiration in this work.

Honey bees daily life cycle involves collection of good nest or searching for nectar in neighbouring environment. Although the individual insect has limited ability, collectively they can perform complex tasks using their self-organising behaviour without any recognised interaction between them. The nest or nectar collection activity can be expressed in following four categories: *i)* Scouting/foraging; *ii)* Pattern formation; *iii)* Pattern recognition; and *iv)* Decision making behaviour. Recently, roboticists have shown keen interest on developing new models inspired by the honeybee waggle dance due to its ability for passive communication, which helps to improve adaptive robotic behaviour and avoids complex multi-point (wireless) communication among robots[28].

In this paper, we are particularly interested in mimicking honeybees' waggle dance as a form of passive action recognition within MRS. Although there are a number of papers available in the literature that explain the pattern formation [15, 44] and pattern recognition behaviour [17] of honey bee waggle dance, to the best of authors' knowledge, its application in robot communication is largely unexplored. The proposed approach considers the design of a multi robot system, where a scouting (leading) robot generates a behavioural pattern by body movement [23] while follower robots recognise and decode the pattern without the need to implicitly or explicitly exchange information among themselves. The main contributions of this paper are threefold:

1. Mimicking honey bees' waggle dance in multi-robot system which includes,
    - Simple and complex *pattern formation* resembling scouting/foraging behaviour and
    - *Pattern recognition* by observing and recognising previously formed patterns (a behaviour of follower bees).
2. Simulating our proposed system using Robot Operating System (ROS[1]).
3. Prototyping the MRS using a group of Unmanned Aerial Vehicles (UAV) (*i.e.,* Parrot AR. Drones[2]).

The paper is organised as follows: background and related work are described in Section 2. Details of the overall system and experimental set up are discussed in Section 3 following the methodology on pattern formation and pattern recognition in Section 4. Results obtained from simulations and real environment are reported and discussed in Section 5 followed by concluding remarks and future work in Section 7.

2. Background and Related Work

Swarm intelligence and biologically inspired computation, especially in robotic applications, have gained significant attraction from researchers in recent years [5]. Many systems

[1] `http://www.ros.org/`

[2] `http://ardrone2.parrot.com/`

have been proposed in the literature that mimic the collective behaviour of insects or animals to perform complex tasks using a group of simple robots (often referred as agents). Many bio-inspired algorithms, such as ant colony algorithm, firefly algorithm, bee algorithm and particle swarm optimisation have been applied in various areas of science and engineering research [42, 11, 52]. Communication in a swarm is extremely important because, robots must share their information to achieve a task. Increasing the number of robots in a MRS decreases the amount of time needed to complete a given task. However, this may not be always true in a practical scenario as multiple robots struggle to speed up the task due to limited communication bandwidth. As a result, the performance of the system degrades as more robots are employed. Thus, we need a good communication system to flow the information uninterruptedly. In this section, we shall discuss about various types of communication proposed in the literature and used in MRS, followed by an overview of honeybees' life-cycle and waggle dance in the context of this paper.

2.1. Multi-robot communications

The term robotics consists of sensing information from the environment, understanding the main features, modifying them with their requirements and then acting on the environment. The main objective in MRS is to achieve the final goal through inter-robot interactions. To achieve this goal, robots require information about their teammates and the environments. According to Parker [37], this information can be acquired by three common techniques: *a)* explicit communication, *b)* implicit communication and *c)* passive action recognition.

2.1.1. Explicit communication

Explicit communication is based on the intentionally transmitting and receiving information via some type of protocol or language as a medium. This is always intentional and the robots are completely aware of it. An example is human's interaction with each other using spoken languages. Deploying this type of communication in MRS always requires some medium, *e.g.,* radio, Ethernet or wireless. However, the communication medium cannot always be shared, therefore it is necessary for the robots to obtain exclusive access to them. The problem of communication medium sharing is often associated with bandwidth limitation. Rekleitis *et al.* [41] examined the problem of multi-robot coverage path planning for a team of robots with limited communication, where the robots operate under the restriction that communication between two robots is only available when they are within the line of sight of each other. In comparison, explicit communication is less robust than implicit communication as communication desires to be transmitted and received in a separate procedure [1]. The authors developed an autonomous and decentralised robot system called ACTRESS. Each robotic agent is able to act autonomously and can interact with other agents. As ACTRESS is a decentralised system, it does not need a supervisor. The authors relied on explicit communication to exchange information using a wireless communication framework. In this case, wireless communication is advantageous in terms of mobility of the autonomous mobile robots. Multiple implementation of one-to-one communication permits one-on-n communication. However, with the increase in number, the communication system

reaches its bottleneck, taking longer to execute the tasks, even though some agents are not active at any given instances. Parker [36] proposed a mobile robotic system (ALLIANCE) to tackle dangerous tasks to reduce the risk for humans. The author developed a behaviour-based architecture with a team of mobile robots performing hazardous toxic waste clean-up tasks. The group of robots were designed to be fault tolerant, reliable and adaptive in nature. An explicit broadcast communication system was developed to form interactions between individual robots. Such communication mechanism allows robots to inform other members of the team about its current activities without establishing any two-way conversations. Although the robots do not entirely rely on sensing through the world, the communication medium is not guaranteed to be available. In some cases, when each robot broadcasts a statement of its current action, other robots may choose to listen or ignore.

In a previous work [10], we proposed a communication architecture for the Robotic Darwinian Particle Swarm Optimisation (RDPSO) with the intent to overcome the Mobile Ad-Hoc Network (MANET) constraints. The problem was described as having a population of N robots, divided into several swarms of NS robots, wherein each robot would be both an exploring agent of the environment and a mobile node of a MANET that performs packet forwarding according to a paradigm of multi-hop communication. The goal was to ensure that robots would explore an unknown environment, while ensuring that the MANET would remain connected. The dynamics of the communication data packet structure shared between robots was described and a set of simple communication rules was proposed in order to reduce the communication overhead within swarms of robots. Several experimental results with up to fifteen real robots in a large scenario clearly allowed us to observe the advantages of such an optimised strategy regarding the scalability of the algorithm, thus paving the way for future swarm applications of hundreds or thousands of robots.

The use of explicit communication can ensure the accuracy of the exchange of information between robots. However, the communication load of a system will increase as the number of robots increases. This may cause a decrease in system performance or else lead to an overall system failure in extreme cases.

2.1.2. Implicit communication

This type of communication uses the environment as a communication medium. This should be achieved by embedding different kinds of sensors in the robot. Implicit communication can be categorised into two types: active implicit communication (*e.g.,* interaction via the environment) and passive implicit communication (*e.g.,* interaction via sensing). Active implicit communication refers that the robots communicate by collecting the information of others in the environment. Passive implicit communication refers that the robots communicate by perceiving a change of environment through the use of sensors [54].

According to Beckers *et al.* [4], robots that use implicit communication, interact with each other through the effects they leave on the environment. An example is ant foraging behaviour where some ants lay pheromone on the way of their food foraging. Using the smell of pheromone trails they exchange the information of food source and the shortest path of the food to the hive. Authors presented several experiments, where a group of randomly distributed robots were trying to build a single cluster. The robots were completely

autonomous and independent. They interacted with each other and with the experimental environment using an indirect method of communication mimicking stigmergy. This type of communication needs no encoding or decoding, no knowledge of place, no memory and it is not transient. In brief, the robots only react to the local configuration of the environment. However, with the increasing number of robots, the number of interaction also increases, which is time consuming and sometimes destroys the existing cluster as the robots have no knowledge to detect whether the task was completed or not. Kube and Zhang [27] constructed a multi robot system to perform a cooperative task (*e.g.,* box pushing) without any centralised control or explicit communication system. The authors have designed an algorithm solely based on the local information of the environment. The advantage of this type of communication is that one can increase the number of robots for certain tasks. Yet, increasing the number of robots decreases the response time and as a result, the robots may be unable to complete the collective task.

Nevertheless, as discussed in Section 1 neither explicit communication nor implicit communication architectures are suitable in all adverse environments or all application scenarios, *e.g.,* security applications. In these particular situations, we advocate passive action recognition as an alternative reliable communication technique.

2.1.3. Passive action recognition

In a different approach, according to Huber and Durfee [22], considered the use of passive action recognition, where robots should be able to observe the behaviour and actions of their teammates, *e.g.,* their body languages. Robots communicate among themselves without any medium or language. For such communication, robots should be able to recognise other robots' behaviour and identify what their actions mean. The authors developed a concept of coordination through observation. An agent coordinates its behaviour using plan recognition techniques to obtain the coordination information. This is useful in dangerous situations where the agent cannot communicate its plans and goals to other teammates. However, there are some costs associated with such recognition, specially the uncertainty due to imperfect observation and inference. Novitzky *et al.* [34] focused on the cooperative interaction in a MRS using the ability to understand the behaviour and action of other robots. The authors defined a task performed by an autonomous robot using its own dance behaviour, called *Infinity Pattern*, to indicate the location of a mine-like object (MLO). An autonomous surface vehicle (ASV) using behaviour recognition technique decoded the *Infinity Pattern* and recognised the location of MLO. Though it is advantageous to predict team members' behaviour, which improves the overall performances, this is a time consuming technique.

Ballagi *et al.* [3] introduced an action selection method for a multi robot task sharing problem. The authors proposed to use a fuzzy signature evaluation and decision-making system for intention guessing and efficient action selection. A code book was built to allow robots recognising the situation and taking action accordingly. In this case, two robots should possess the same part of the code book, otherwise, the information to be transmitted might be distorted and may end up with a deadlock combination. Ghosh and Marshall [16] proposed a suitable model for collective decision-making method in a swarm of robots, where each robot has to choose the better option among several alternatives having different profits.

One of the drawbacks of this model is the use of perfect communication, which is unrealistic in a real-world context.

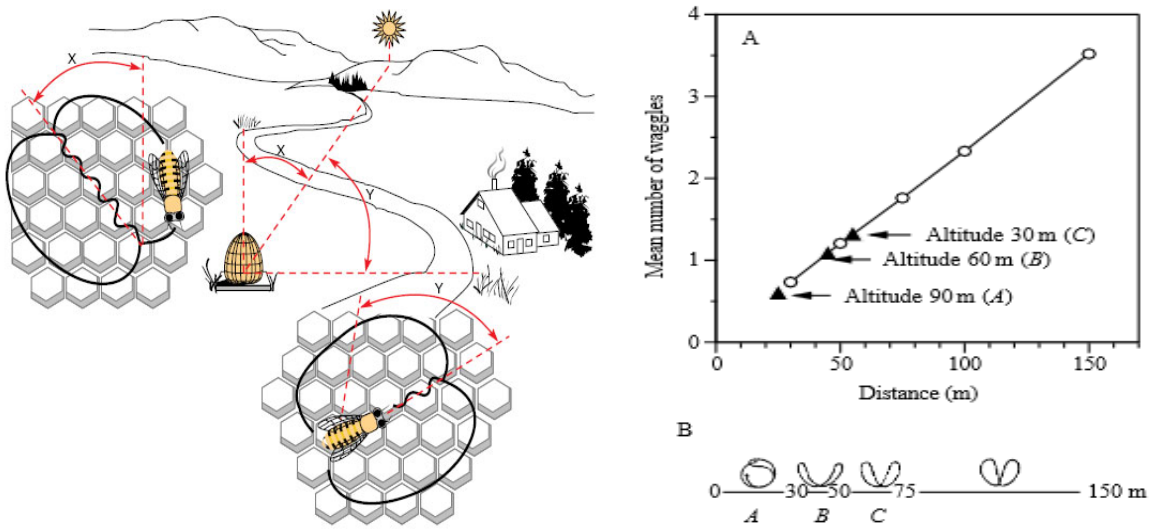
Although passive action recognition embraces complex mechanisms and is a major research topic in itself, this paper presents a simplistic, and still reliable way to passively communicate in a MRS by getting inspiration from honeybee's life-cycle and waggle dance.

2.2. Honeybees' life-cycle and waggle dance

Honeybees' life-cycle is a fascinating biologically inspired computation endeavour due to their socially complex colonial culture. Although the individual insects have limited ability, collectively they can perform complex tasks using their self-organising behaviour without any implicit or explicit communication among themselves. This is achieved by their body movement known as *Waggle Dance* [53]. The communication involves pattern formation depicted in the waggle dance and pattern recognition, which is decoding the hidden information within the dance actions.

Nectar collection is a major activity in honeybees' daily lives and involves a group of worker bees. There are two types of worker bees present in a beehive depending on their work load: *a)* forager or scout bee and *b)* follower bee. A forager honeybee randomly visits different flower sites searching for promising nest site and food resource (*i.e.,* nectar). The foraging behaviour depends on the environmental temperature and time of the year. They rely on direction of the sun, polarised light pattern (if it is cloudy day), smell of flowers and earth's magnetic fields. They start flying to a particular direction with respect to the azimuth angle of the sun to the hive. Janson *et al.* [23] described how does the scouting behaviour of honeybee allow selecting good quality nest/food sites that is far away. Once nectar was collected, the forager bee returns to the hive, and attempts to attract other follower bees for the visited site. To convince other bees and communicate the site information, the forager bee performs the *Waggle Dance*. Two types of dance pattern are generally observed: if food source is too close, they make a round shape dance which indicates only direction and whereas small eight like shape represents a far away food source. The waggle dance is correlated with the direction and the distance of the suitable nectar resource to the hive. Dance direction depends on the azimuth angle of the sun to the food source with respect to gravity. Figure 1a) shows how does the direction of waggle dance depends on the azimuth angle of sun to hive. The forager bee also adjusts her dance accordingly to accommodate any time lapse to the changing direction of the sun. Therefore, the follower bees can accurately move towards the food source although the angle of the sun changes continuously. The duration of the waggle dance and the number of waggle movement changes with the distance are described by Esch and Burns [15] and Seeley *et al.* [44]. Distance vs waggle phase graph is shown in Figure 1b).

Multiple forager bees participate in the dancing competition and, at the same time, they try to attract other follower bees. The follower bees closely observe the forager bee's body movement, dance orientation, vibration of the wings, and duration of the waggle dance. A forager bee should dance more lively and perform longer so that it can communicate the detailed information about existence of a highly reliable and quality food source. The followers decode and recognise the information communicated through the waggle dance



(a) (b)

Figure 1: The waggle dance is correlated with the direction, the distance and the azimuth angle of the sun to the suitable nectar resource. (a) An example of waggle dance direction that depends on the azimuth angle of the sun to the hive ([6]) and (b) Distance vs Waggle phase graph ([15]).

and then take a decision to follow a particular forager bee. The details of the techniques used by follower bees to decode and recognise the information provided by the forager bees through their waggle dance are still largely unknown. Understanding the information is very challenging as the duration of the waggle dance is small and the forager bee is overcrowded by the other follower bees. Therefore, the follower bees may need to encode some meaningful information from the dancing bees as described by Gil and Marco [17]. The communication between forager and follower bees through waggle dance is generally accurate and robust, and thus the motivation of this paper.

The literature presents some papers that are inspired by honey bees' life-cycle or waggle dance [38, 28, 2]. Recently, Landgraph *et al.* [29] recorded and analysed honey bee waggle dance motion trajectories of European honey bees. A set of properties was used as input parameters to model a biometric honey bee robot, such as global parameter, waggle run parameter, return run parameter and intra waggle run parameter. This model has been able to produce trajectories similar to real ones. In this paper, we go a step further by mimicking the waggle dance in multi-robot communication, with the intent to propose an alternative passive action recognition technique, robust to environmental variations.

3. MRoCS design

In order to mimic the honeybees' waggle dance accumulation system, we proposed and developed a novel communication infrastructure for a multi-robot-system using UAVs. This section describes the overall system design and the experimental setup used in this work. The details of the methodology are discussed later in Section 4.

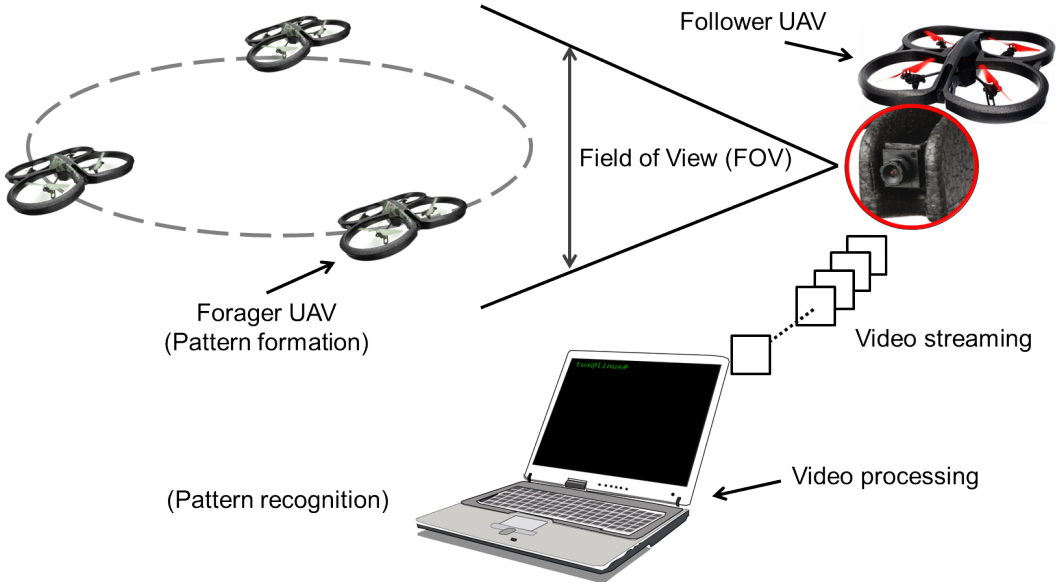


Figure 2: System overview: A forger UAV forms pattern using pre-defined geometrical shapes; The follower UAV captures the patterns using onboard front camera and streams the video wireless to a base computer for further processing; and the patterns are recognised using proposed image / video processing algorithm running at the base computer.

System overview:. Passive action recognition based communication models (*e.g.,* waggle dance), include two distinct phases: *a)* action or pattern formation; and *b)* pattern recognition [7]. In a MRS, the former step can be achieved by forming different shapes using robotic motion of individual agents, while we propose a computer vision based algorithm for the latter step. The proposed approach reveals an effect of communication between robots, similar to broadcasting or, more specifically, geocasting, by updating neighbourhood robots knowledge, which affects their own decision-making and, inevitably, the collective decision-making of the team. The key difference is that no explicit messages are exchanged between robots, making the system more robust, unpredictable and secure. For the design of our system, we chose UAVs as robotic agents due to their flexibility and capability to form patterns in every direction (X, Y and Z-axis). On the other hand, recognising objects or patterns can be achieved using various sensors, including infra-red [31], LiDAR [20], ultra-sound [33] or optical camera sensor [45]. In this paper, we chose the latter one due to its availability with popular UAVs, *e.g.,* AR Drone [25], and the availability of robust computer vision algorithms [49].

An overview of the system is shown in Figure 2. The overall system and the algorithms are developed using ROS (for pattern formation) and MATLAB[3] (for pattern recognition). As the proposed method relies on various software tool-sets and UAV hardware, descriptions for those are necessary in order to understand and justify the proposed work.

[3] `http://www.mathworks.com/products/matlab/`

(a) (b)

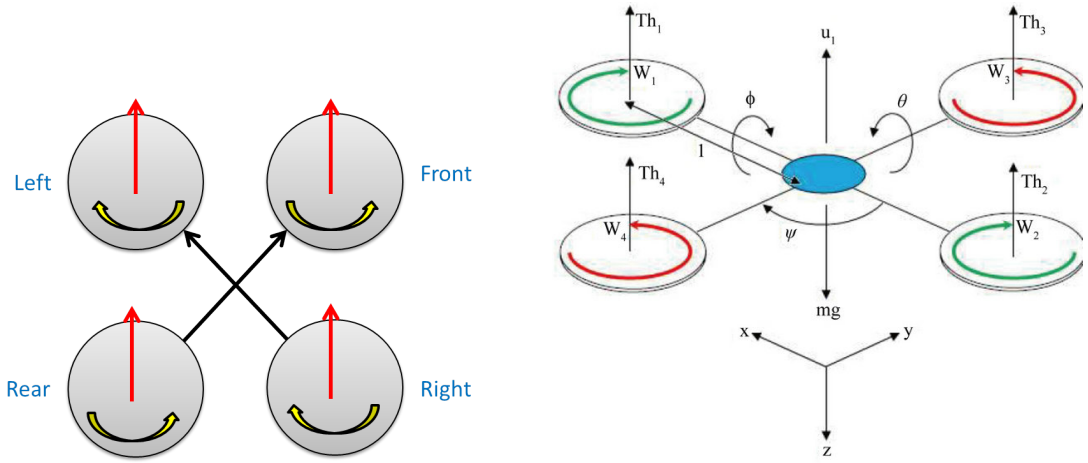
Figure 3: AR.Drone 2.0 configurations for (a) Outdoor navigation (light configuration) and (b) Indoor navigation (protected by external hull).

System requirements:. In designing the system, we built two environmental setups for simulation and real-life implementations. In simulation, we ran all the experiments using *tum_simulator*[4] using virtual UAVs, and Parrot AR Drones were used in real environment. Flight manoeuvres for pattern recognitions were done by series of several *ROS commands*. In real environment, the AR. Drone was controlled by ROS commands through a standard WiFi network between a laptop and the AR Drone. *ardrone_autonomy*[5], a ROS driver that was used to control the paths, altitude and orientation of the AR Drones. OpenCV[6] and MATLAB were used for video streaming and video processing to detect patterns, respectively.

In this work, we chose AR. Drone 2.0, equipped with optical sensors, for two reasons: *a)* it has the capability to mimic forager honeybees since it is lightweight and easily controllable and *b)* accessibility of the onboard 2D RGB sensor which is suitably used for pattern recognition using a novel image processing algorithm. AR. Drone 2.0 was developed by Parrot in 2010, and resembles a toy with sensors, cameras, and basic autonomous behaviours, such as take-off, hovering and landing. According to [25], the AR. Drone has become a popular platform for research and education due to its relatively high performance and low cost. It has been used for object following, position stabilisation and autonomous navigation. When flying outdoor, the AR.Drone 2.0 uses a lighter configuration (as shown in Figure 3a)). When flying indoor, it is protected by external bumpers (as shown in Figure 3b)).

The mechanical structure contains four rotors attached to the four ends. When moving forward, each pair of opposite rotors is turning the same way, as shown in Figure 4. One pair is turning clockwise, while the other anti-clockwise. The AR. Drone manoeuvres by changing Roll (ϕ), Yaw (ψ) and Pitch (θ) value. The AR. Drone yields roll movement by varying left and right rotors speeds in the opposite directions. This allows to go forth and back. Varying front and rear rotors speeds the opposite way, it yields pitch movement. Varying each rotor

[4] `http://wiki.ros.org/tum_simulator`
[5] `http://wiki.ros.org/ardrone_autonomy`
[6] `http://opencv.org/`



(a) (b)

Figure 4: (a) Mechanical movement of an AR. Drone with pairs of opposite rotors, (b) AR. Drone direction of movement control by changing Roll (ϕ), Yaw (ψ) and Pitch (θ) value.

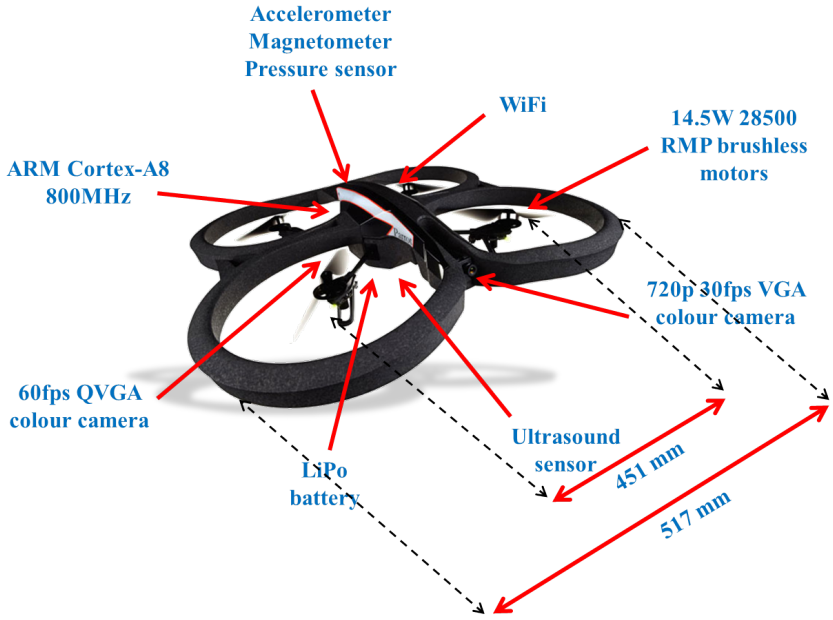


Figure 5: Different sensors and hardware dimensions of AR. Drone 2.0.

pair speed the opposite way, the drone yields yaw movement, which allows turning left and right. With reference to Figure 5, the outdoor drone measures 451 mm by 451 mm, without the protective hull attached, with a weight of 380 gm. The indoor drone is 517 mm by 517 mm, with the protective hull, and its body weighs 420 gm.

One 720 HD-30 fps camera is at the front of the vehicle and one 60 fps bottom facing camera is attached aiming towards the ground to get round speed measurements for auto-

matic hovering and trimming. An ultrasound sensor is attached for altitude measure and vertical speed control. It also encompasses a pressure sensor to measure the altitude at any height. The AR.Drone 2.0 requires a charged 1000mAh, 11.1V LiPo batteries to fly, which allows 12 minutes of flying time. It is powered with four 14.5W 28000 RMP brush-less engines with three phase current controlled by an ARM Cortex A8 processor. It can detect if any of the propeller is blocked, and in such case, stops all engines immediately. It has a master USB port, with a standard USB-A connector. It can communicate over WiFi with external devices, such as the control Linux unit, and can be controlled using iOS and android devices. However, due to the limitations in terms of processing and operating system within AR.Drone's embedded ARM, the high-level behaviour was implanted in the control Linux unit by benefiting from ROS.

ROS is a flexible framework for writing robot software. We used ROS framework to build a communication system between AR. Drone and computer over WiFi network. ROS fuerte version was installed on Ubuntu 12.04 workspace, including *ardrone_autonomy* and OpenCV. ROS is a collection of hardware abstractions, device drivers, libraries, visualisers, message-passing, package management, etc., to develop and create robot applications across a wide variety of robotic platforms. *Ardrone_autonomy* is a ROS driver for Parrot AR. Drone quadcopter. This driver is based on an official AR. Drone SDK version 2.0 and supports both AR. Drone 1.0 and 2.0. *ardrone_autonomy* is used to plan and execute missions by describing the path, altitude and orientation of the drone to follow. Many ROS control commands were generated to manoeuvre the AR. Drone flight. Changing the angular velocity and the linear velocity one can form different types of pattern. For example, flying forward and backward can be done by simply using following commands:

```
fly forward:
rostopic pub -r 10 /cmd_vel geometry_msgs/Twist
'{linear:  {x: 1.0, y: 0.0, z: 0.0}, angular: {x: 0.0,y: 0.0,z: 0.0}}'

fly backward:
rostopic pub -r 10 /cmd_vel geometry_msgs/Twist
'{linear:  {x: -1.0, y: 0.0, z: 0.0}, angular: {x: 0.0,y: 0.0,z: 0.0}}'
```

In this work we used OpenCV to stream real-time video file captured by follower UAV to the computer.

4. Methods

This section describes and discusses the herein proposed methodology. The functional blocks of the overall system are shown in Figure 6 and described in detail in Section 4.1 and Section 4.2.

4.1. Pattern formation by forager UAV

To create patterns using UAVs, we developed stacks using ROS framework, which would then issue series of control commands to the quadcopter's rotors (AR. Drone here). We

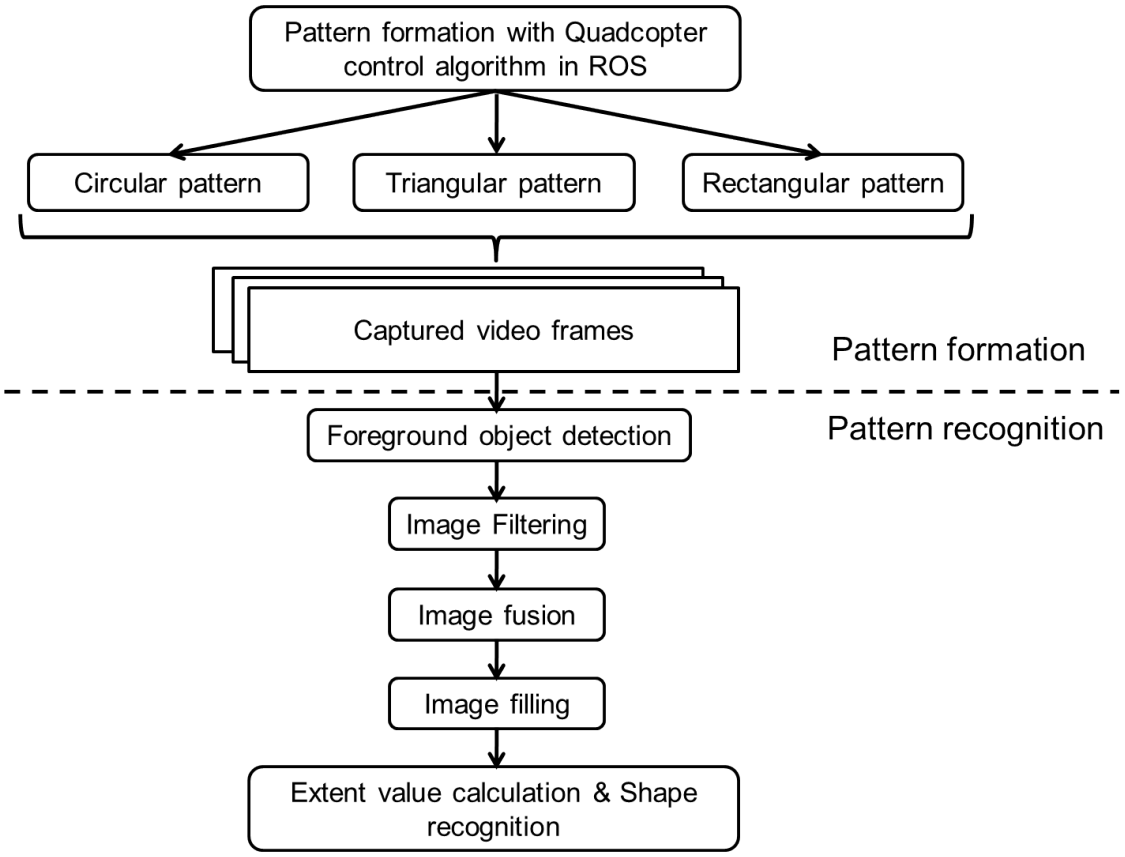
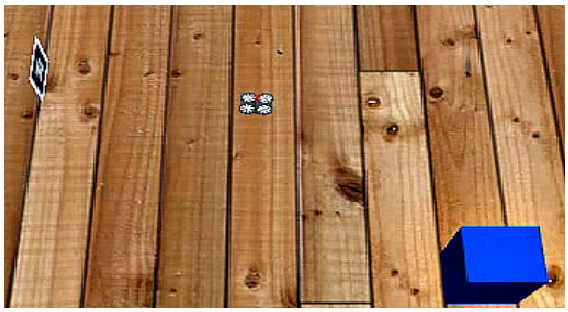


Figure 6: Functional block diagram of the overall system.

selected three basic geometric shapes as they are easy to build and recognise: *1)* circle; *2)* triangle; and *3)* rectangle. In all three cases, the UAV flew in the air and autonomously formed those shapes in the XY plane represented in Figure 4(b). The ROS framework was used to iteratively manoeuvre the directions so as to determine the adequate predefined flight path. The size of these patterns was varied by increasing or decreasing the values of the parametric rotor combinations. Due to the imperfect synchronisation among the rotors, slight instability of the flight paths were observed. To address this issue, the experiments were repeated 24 times for each pattern (12 in simulations and 12 in real scenarios), to ensure the robustness of the results (Section 5). Initially, all the patterns were created in *tum_simulator*, following a similar test in real environment. In the simulator, we ran all experiments in two different environmental scenarios, as shown in Figure 7a. In real life, we performed all the experiments in the laboratory, as well as in a large astrodome pitch (shown in Figure 7b).

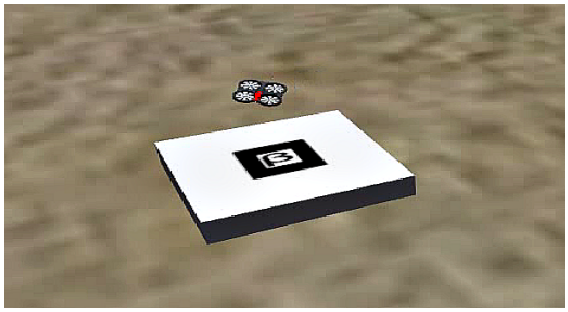
4.2. Pattern recognition by follower UAV

To implement the pattern recognition step, we capture the videos of the forager UAVs forming pattern using a camera attached to the follower UAV and stream it for further processing in MATLAB. It is worth noting that, any onboard real-time processing of the



3boxes_room environment.

Inside laboratory environment.



outdoor_flight environment.

(a) tum_simulator environment.

Inside astrodome pitch.

(b) Real environment.

Figure 7: Example of various environments used in this work.

system is not within the scope of this paper. We are particularly interested in proposing the passive action recognition communication model with a proof-of-concept implemented in both simulated and real UAVs. The main objectives of the pattern recognition step are to locate, identify, and categorise patterns in images and videos. To that end, the modelling of object motion and appearance changes are necessary for video-based object recognition. Existing algorithms either rely on appearance model, which is either fixed or rapidly changing, or on the motion model. Often, background clutter or illumination changes make the recognition or tracking problem extremely challenging. Further information fundamentals of image processing and computer vision algorithms can be found in [18] and [50].

In this work, the goal is to identify the region of interest (the pattern formed by forager UAV) in each frame of the video, and track it efficiently, so as to identify a pattern created by the forager UAV. Finally, the patterns can be recognised using machine learning or template matching algorithms. The proposed pattern detection and recognition process is developed using five functional processes. A summary of the complete algorithmic flow of these functions is shown in Algorithm 1 and described later in the text. It is worth noting that, in this work we considered that the follower drone is stationary while recognising the patterns. In the real life scenario, the follower drone can either be static or moving depending on the circumstances. The pattern recognitions of a moving target from a moving drone using image processing techniques are far more challenging problem and are outside the

Input: Video sequence ($\mathcal{V}$) with waggle dance pattern by foraging bees;
Initialise foreground detector;

Read first frame: $\mathcal{F} = videoread(\mathcal{V})$;
Foreground detection: $\mathcal{F}_{mask} = foreGroundDetector(\mathcal{F})$;
Median filtering with 7×7 window: $\mathcal{F}_{filt} = meadianfilter(\mathcal{F}_{mask}, [7 \times 7])$;
Initialise frame fusion: $\mathcal{F}_{fuse} = \mathcal{F}_{filt}$;

repeat
  Input: Receive next frame;
  $\mathcal{F} = videoread(\mathcal{V})$;
  $\mathcal{F}_{mask} = foreGroundDetector(\mathcal{F})$;
  $\mathcal{F}_{filt} = meadianfilter(\mathcal{F}_{mask}, [7 \times 7])$;
  $\mathcal{F}_{fuse} = OR(\mathcal{F}_{fuse}, \mathcal{F}_{filt})$;
until *end of sequence*;

Region filling of the flight path: $I_{shape} = imfill(\mathcal{F}_{fuse}, \text{'Holes'})$;
Region detection: $\mathcal{R} = regionprop(I_{shape}, \text{'all'})$;
Calculate extent value: $\mathcal{R}_{extent} = extent(\mathcal{R})$;

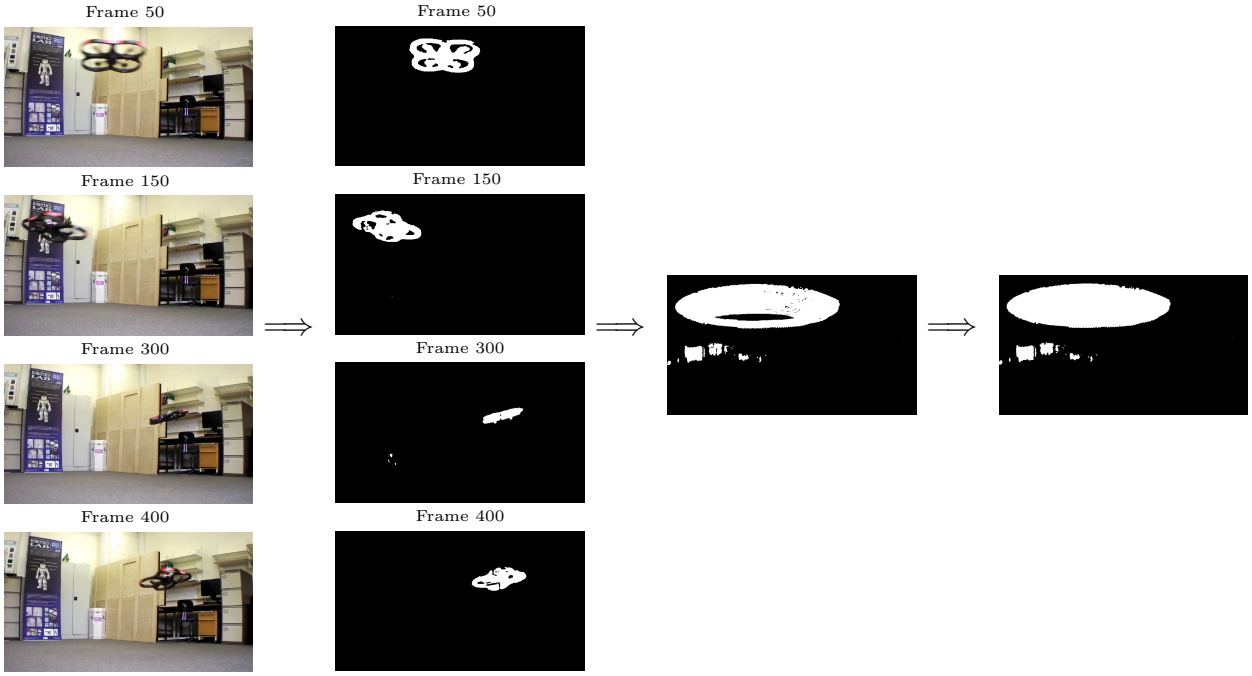
Pattern (P) detection:
if $0.45 < \mathcal{R}_{extent} < 0.65$ **then**
  $P = Triangle$;
else if $0.65 < \mathcal{R}_{extent} < 0.82$ **then**
  $P = Circle$;
else if $0.82 < \mathcal{R}_{extent} < 1.0$ **then**
  $P = Rectangle$;
else
  $P = No\ Pattern$;

Algorithm 1: Summary of the pattern recognition process by follower bees.

scope of this work.

4.2.1. Foreground object detection

Foreground detection is often considered as one of the major tasks in computer vision research domain and it aims to segment moving foreground objects in image sequences. Algorithmically, a foreground detector system compares the current video frame with either a predefined background model or adaptively learned background model [39]. The individual pixels are compared to determine whether that is part of the background or the foreground. A foreground mask is then computed using a background subtraction model which segments foreground objects in an image taken from a stationary camera. In a dynamic scene, where the background is unknown to the system, first the algorithm tries to model it by considering the first frame as the entire background of the image. As a following step, it computes any changes found in the background. However, this is challenging and difficult as the scene may contain shapes, shadows and moving objects. We chose a state-of-the-art adaptive foreground detection model [26, 48] available from computer vision toolbox in MATLAB. In this paper, the foreground detector is used to mask the forager UAV as foreground object in each frames in the sequence. It returns binary values where the background is represented by 0 and the foreground by 1. An example of this process is shown in Figure 8a and Figure 8b.



(a) Original video frame

(b) Foreground object after filtering noises.

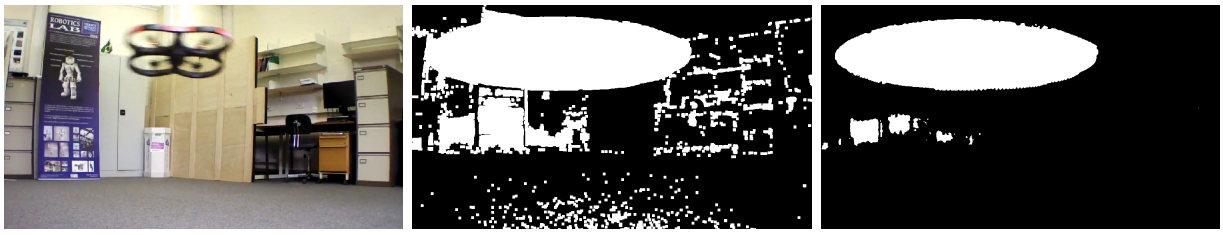
(c) Flight path detection using image fusion.

(d) Image region filling for shape recognition.

Figure 8: Different stages of pattern recognition.

4.2.2. Image filtering

Digital images are prone to a variety of resulting noises [18]. Noise is the result of errors in the image acquisition process that result in pixel values that do not reflect the true intensities of the real scene. Once the object foreground has been recognised, the next step considers image filtering. By definition, the filtered value of every pixel of an image is the sum of the weighted intensity value of the current pixel and neighbouring pixel values. These weights are called filter matrix [18]. In simulation, commonly used mean or average filters are used to smooth the foreground objects. In average filtering, each output pixel is set to an average of the pixel values in the neighbourhood of the corresponding input pixel. This helps to generate a smooth and continuous flight path in the later stage of the functional blocks. However, in real environments (Figure 9a), average filter is not effective enough to reduce the noise. Vibration of rotors and shadows created by UAV's movement affects the light intensity, as shown in Figure 9b. As a result, the foreground is incorrectly detected which introduces clutter like noises and makes it challenging to determine the flight path. In this case, we used median filtering, which is similar to an averaging filter, but instead of averaging the value of an output pixel, the median of the neighbourhood pixels is considered. Median filtering is, therefore, better able to remove these outliers without reducing the sharpness of the image [30]. This is clearly visible in our result as shown in Figure 9c.



(a) Example frame of a real environment. (b) Processed pattern with average filter. (c) Processed pattern with median filter.

Figure 9: Example of applying average filter and median filter in real environment. Vibration of rotors and shadows created by UAV's movement, affects the light intensity and generates noise. Unlike an average filter, a median filter effectively removed the background clutter.

4.2.3. Image fusion

Image fusion is the process of combining relevant information from two or more images into a single image [8]. The resulting image will be more informative than any of the input images. In this work, we intend to recognise a single pattern from a sequence of images (video). Therefore, fusion of the foreground images is a logical choice which effectively creates a single image of the flight path. The fused image is then used for the geometric shape recognition. There are many sophisticated and complex image fusion algorithms available. However, in our case, the filtered foreground masks are binary images. Therefore, we considered simple 'OR'-ing between successive images as the most effective and computationally efficient fusion operation, as shown in Figure 8c.

4.2.4. Image filling

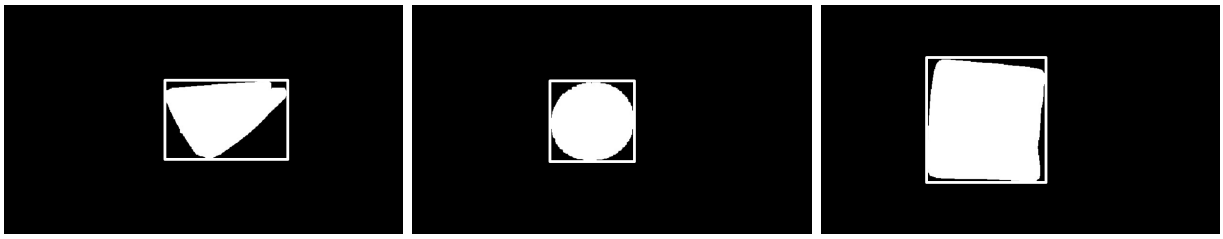
The region filling, or hole filling, is a technique to fill target region, or hole, in the image [47]. This is particularly useful for image segmentation and shape recognition applications. A region, or hole, is defined as a background region surrounded by a connected border of foreground pixels. This is useful for our purpose as this eases the geometric shape recognition problem as shown in Figure 8d.

4.2.5. Extent value calculation and shape recognition

After processing the image sequences, we obtained a fused image region with 2D geometrical shapes. In this functional block, we aim to measure the properties of this region in order to quantify the shape (region of interest). Our interest is to calculate the extent value as it helps to identify the geometric shape of the region and is used to recognise shapes in image processing applications [40]. The extent value of a region is calculated by finding the ratio between the area of the region and the area of its bounding box:

$$\text{Extent} = \frac{\text{Area of the object}}{\text{Area of bounding box}}. \tag{1}$$

For circles, this value is fixed at 0.78, irrespective of the radius. The corresponding value for rectangles and squares leans towards 1, provided the sides are parallel to the axes and



Extent value= 0.597 Extent value= 0.809 Extent value= 0.903

Figure 10: Different geometric shapes with bounding box and their extent values.

the bounding box and sides overlap. Triangles are tricky as it is difficult to find the correct inclinations of the three vertices. In an ideal case, the Extent value of a triangle must always be less than or near to 0.5. These values are true for the cases where the observations are orthogonal to the shape. However, in this work we faced two issues: *1)* due to instability in rotor synchronisation, the UAV flight paths do not follow an ideal circular, triangular or rectangular path and often generate a slightly curved path towards the corners; *2)* viewing angle of the follower UAV may not be orthogonal to the patterns formed by forager UAVs. Therefore, we have empirically extended the ranges of the extent values to detect the correct pattern. The values were defined as $(0.45 - 0.65)$ for a triangle, $(0.65 - 0.82)$ for a circle and $(0.82 - 1.0)$ for a rectangle. Examples of the different geometric shapes with bounding box and their associated extent values from our experiments are shown in Figure 10.

5. Experimental results

Initially, all the experiments were carried out in tum_simulator for two different scenarios: *1)* outdoor environment; and *2)* indoor environment (refer Figure 7a). Afterwards, we performed all the experiments in real environment in two other different scenarios: *1)* lab environment; and *2)* astrodome environment, as shown in Figure 7b. For statistical significance, we have repeated the experiments for at least twelve times for each pattern individually in simulation and real environment, generating combined 72 experimental sequences. The patterns, formed in the experiments are shown in Figure 11 and Figure 12 and the results of the experiments are shown in Figure 13, for triangular, circular and rectangular flight paths, respectively. In Figure 11 and Figure 12, first three rows in each figure represent different image frames from three different video files of circular, triangular and rectangular patterns following a column to show captured flight path after image fusion. Finally processed 2D shapes after image filling operation are shown in *Row 5*. The extent values calculated from all 72 sequences are shown in Figure 13 where *Columns 1* and *2* are for simulation (36 sequences) and real (36 sequences) environments, respectively. In each graph, the patterns are grouped by sequences with triangular (Sequence 1-12), circular (Sequence 13-24) and rectangular (Sequence 25-36) patterns. The x-axis of the graphs represents the sequence number and the y-axis shows the calculated extent value after pattern recognition steps described in Section 4.2. Different shades of grey are overlaid to show the ranges of

extent values for correct pattern detection. It is evident from the graphs that simulation results are more consistent (with little variation) over real environment. This can also be observed at various stages of pattern recognition shown in Figure 11 and in Figure 12.

In simulation, the experiments showed promising results for all three patterns. The geometrical shapes were correctly identified 11 times out of 12 experimental run. The extent values were most stable for circular flight path, whereas the rectangular and triangular patterns displayed a wider range of extent values. In real environment, triangular patterns show more promising results followed by the circular patterns. Rectangular patterns exhibit relatively more errors than other two patterns. A statistical analysis is presented in Section 6 followed by an in-depth discussion of the results obtained.

6. Analysis and Discussion

In order to statistically evaluate, firstly we generated confusion matrixes which were then used to compute sensitivity and precision. We briefly described the results against each matrix.

A confusion matrix contains information about actual and predicted classifications done by a classification system. Each column of the matrix represents the instances in a predicted class, while each row represents the instances in an actual class. Performance of such systems is commonly evaluated using the data in the matrix. The confusion matrices for our experiments for three different geometric patterns are shown in Figure 14a and in Figure 14b. Higher values in the diagonals of the confusion matrix represent a successful outcome of our experiments.

We have measured the statistical performances such as *sensitivity* and *precision*, generally known as classification functions in statistics. The measurements are done by calculating the *a)* number of true positive values (TP), *b)* number of false positive values (FP), *c)* number of false negative values (FN) and *d)* number of true negative values (TN). Sensitivity measures the proportion of actual positives which are correctly identified. Mathematically, this can be expressed as:

$$Sensitivity = \frac{\mathrm{TP}}{(\mathrm{TP} + \mathrm{FN})}. \tag{2}$$

The precision of a measurement system is referred to how close the estimates are from expected value under unchanged conditions. Precision or positive predictive value is defined as the proportion of the true positives against all the positive results (both TP and FP). Mathematically, this can be expressed as:

$$Precision = \frac{\mathrm{TP}}{(\mathrm{TP} + \mathrm{FP})}. \tag{3}$$

Sensitivity and precision analysis are appropriate here as we measure the performance of the recognised patterns. The results are shown in Table 1 and in Table 2 for simulation and real environment, respectively.

Sensitivity is basically how good a test is at finding the pattern if it is there. It is a measure of how often the test correctly identifies the actual pattern, among all the experiments

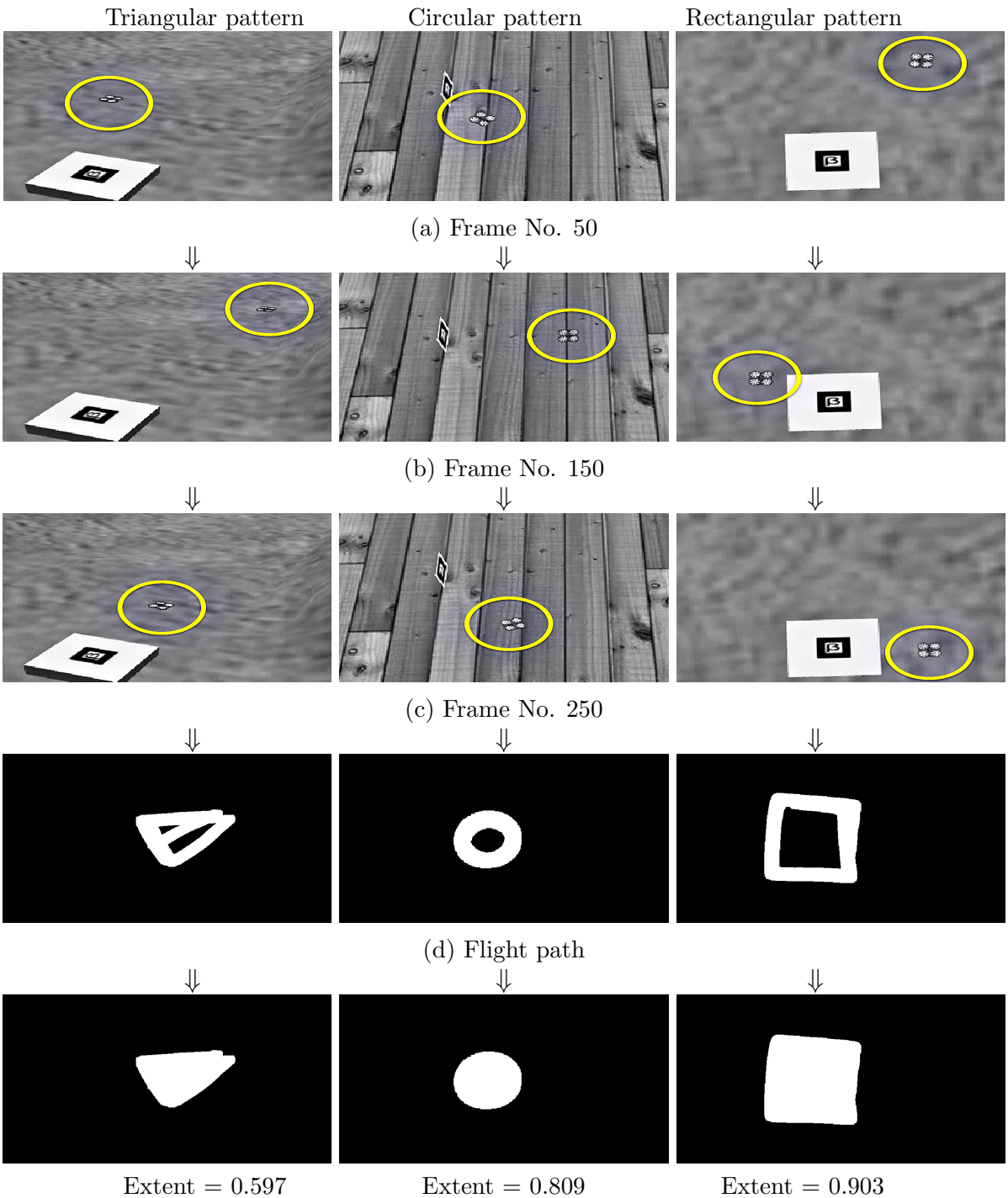


Figure 11: Different stages of pattern recognition in simulation environment. *Rows 1-3* show example frames from the sequences, *row 4* shows the flight path captured by applying foreground detection and image fusion, and *row 5* represents the shapes after image region filling and corresponding extent values. *Columns 1, 2 & 3* represent triangular, circular and rectangular pattern, respectively.

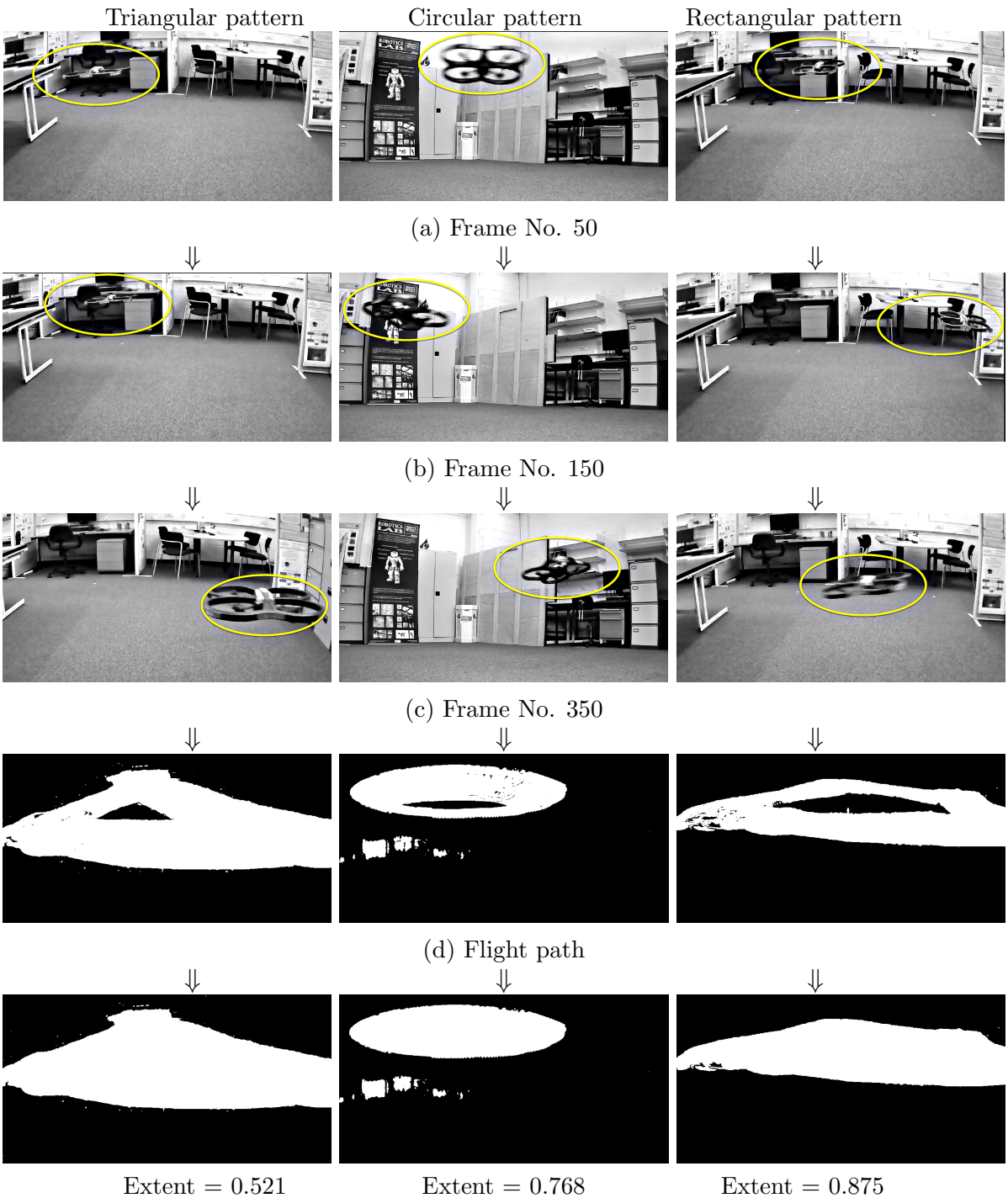


Figure 12: Different stages of pattern recognition in real environment. *Rows 1-3* show example frames from the sequences, *row 4* shows the flight path captured by applying foreground detection and image fusion, and *row 5* represents the shapes after image region filling and corresponding extent values. *Columns 1, 2 & 3* represent triangular, circular and rectangular pattern, respectively.

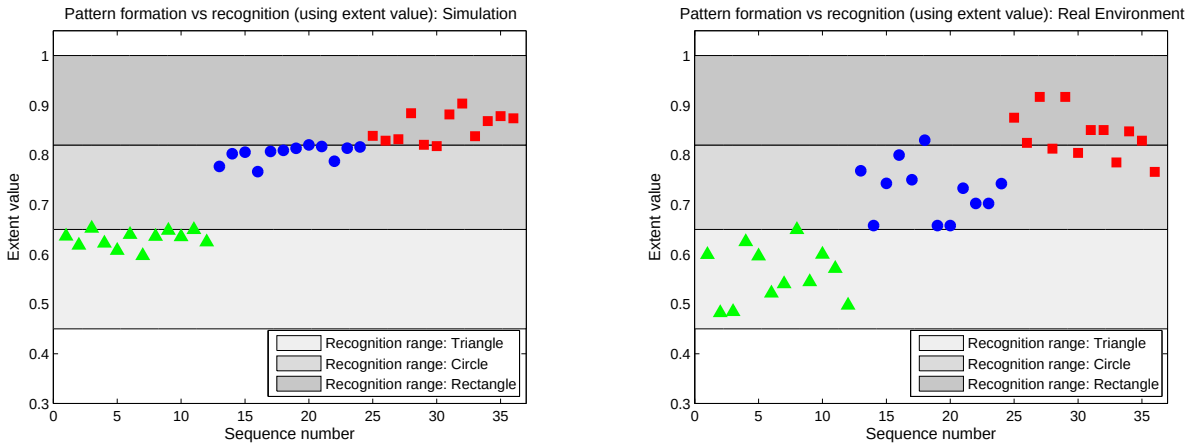


Figure 13: Extent value for three different pattern formations and their corresponding detected pattern in simulation (*column 1*) and real (*column 2*) environment.

		Predicted Class		
		Circle	Rectangle	Triangle
Actual Class	Circle	11	1	0
	Rectangle	1	11	0
	Triangle	1	0	11

(a) Simulation environment.

		Predicted Class		
		Circle	Rectangle	Triangle
Actual Class	Circle	11	1	0
	Rectangle	4	8	0
	Triangle	0	0	12

(b) Real environment.

Figure 14: Confusion matrix of experimental results. Higher values in the diagonals represent a successful outcome of our experiments.

performed. Precision is how close the measured values are to each other. By determining precision values we can realise the range of the extent values of the correctly identified patterns. Higher values of both *Sensitivity* and *Precision* represent a stable and robust system. These measures for our experiments are shown in Table 1 and in Table 2. In the simulation, more than 92% sensitivity is reported for all patterns whereas the precision is recorded as more than 84% and in the real environment, sensitivity is more than 67% whereas the precision is more than 73%. Therefore, it is safe to say that our recognition algorithm is efficient and robust enough in the simulator environment as well as in the real environment. However, it is also worth noting that the success of pattern formation depends on the stability of the rotors of UAVs whereas the pattern recognition algorithm considerably relies on the viewing angle of the recorded video streamed by the follower AR. Drone, illumination changes, noise etc.

Additionally during pattern formation, AR. Drones are not very stable when it hovers close to the wall or the floor of the laboratory. AR Drone has four propellers with diagonally

		TP	FP	FN	TN	Sensitivity	Precision
Class	Circle	11	2	1	22	**0.92**	**0.84**
	Rectangle	11	1	1	23	**0.92**	**0.92**
	Triangle	11	0	1	24	**0.92**	**1.00**

Table 1: Sensitivity and precision in simulation environment. Higher values of both Sensitivity and Precision represent a stable and robust system in the simulation environment.

		TP	FP	FN	TN	Sensitivity	Precision
Class	Circle	11	4	1	20	**0.92**	**0.73**
	Rectangle	8	1	4	23	**0.67**	**0.89**
	Triangle	12	0	0	24	**1.00**	**1.00**

Table 2: Sensitivity and precision in real environment. Higher values of both Sensitivity and Precision exhibit a stable and robust system in real environment.

same directional rotation. When they hover in the air, the propeller blades produce forces that create propulsion pushing, the AR. Drone from the wall or floor. As a remedy we chose environments that have larger open space and height such as *astrodome* to avoid such instability. Detecting pattern of an object in a cluttered environment is a highly challenging problem. Lighting, contrast and poor viewing angle can often make it difficult to distinguish actual pattern from other random clutter. Examples of clutter and lighting conditions creating noises are evident in our experiments. To address such problem we used a median filter replacing an average filter during the filtering stages of the pattern recognition process (as described in Section 4.2.2).

It is evident from the results that, especially in real life scenarios, rectangular patterns are relatively difficult to recognise when compared to the other two patterns. This is due to the viewing angle of the follower UAV. In this case, the lines closer to the camera appear larger than the lines far away from the camera. Therefore, the rectangles appear to be trapezium on a 2D image plane. Similarly, circles are often represented by elliptical shapes. This is a classical multiple view geometry problem in computer vision community [19] and can be solved by geometric triangulation. This is outside the scope of our current work and we aim to address it in future. Alternatively, one can also consider either view invariant shapes or a viewing angle orthogonal to the flight paths of the forager UAVs.

7. Conclusions

In this work we proposed and designed a novel swarm robotic communication system. The system mimics the honey bees' waggle dance, which is a form of passive action recognition technique. Two scenarios were proposed to represent various stages of the waggle dance:

pattern formation of scouting or forager bees (to convey information to other follower bees about the source of food/nest) and *pattern recognition of follower bees* (to recognise and decode the observed pattern by the follower bees). These stages were first designed, implemented and verified on simulation and later realised on a group of Parrot AR. Drones. We considered scenarios with three fundamental geometric patterns: circle, triangle and rectangle. In recognising the patterns, an image processing base algorithm was proposed to track and identify flight paths (patterns) generated by forager UAVs.

During the experiments, the pattern recognition process successfully reported 92% and 67% or more *sensitivity* and more than 84% and 73% *precision* value for three different patterns in simulation and the real environment, respectively. The outcome of this work can be beneficial in the circumstances where traditional communication systems (implicit and explicit communication) often fail. Communication based on passive action recognition is very useful as it is more robust to environmental changes where teams of robots must adapt by continually forming and reforming their swarm behaviour. The proposed model is also useful in energy constrained scenarios as the robots do not need to use any active communication channels *e.g.,* wireless.

Although three fundamental geometric patterns were used in this work, this can be further extended to many complex behavioural patterns of real life honeybees. In the future, we aim to extend the algorithm for complex patterns that represent both directional and the distance information to closely mimic communications of the honeybee waggle dance. We also intend to perform several experiments in various cluttered environments with variable illumination. This is a challenging problem as robust foreground object segmentation via background modelling algorithms is difficult in a real life outdoor environment with unknown clutters and changing lighting conditions. Current work considers formation and recognition of the patterns in 2D plane, while we consider future work by generating 3D patterns which can be captured by a depth camera (along with RGB camera). This will improve the robustness of pattern formation and recognition process. One of the motivations of future work is to develop a group of mobile robots that can operate robustly and fully autonomously in a real world scenarios. An on-board processing unit can be developed to perform all sensing and computation work, making the system independent of any remote base station. Additionally, high level navigation using Simultaneous Localisation and Mapping (SLAM) techniques [32], computer vision and control algorithms can add more automaticity in similar MRS.

Acknowledgements

We acknowledge the support of the School of Engineering and Physical Sciences, Heriot-Watt University, Edinburgh through 'Remission of Fees' and the School of Mathematical and Computer Sciences, Heriot-Watt University, Edinburgh through a visiting researcher fellowship. The authors also thank Deepayan Bhowmik for his feedback particularly in the image processing.

References

[1] Asama, H., Ozaki, K., Matsumoto, A., Ishida, Y., Endo, I., 1992. Development of task assignment system using communication for multiple autonomous robots. Journal of Robotics and Mechatronics 4 (2), 122–127.

[2] Bailis, P., Nagpal, R., Werfel, J., 2010. Positional communication and private information in honeybee foraging models. In: Swarm Intelligence. Springer, pp. 263–274.

[3] Ballagi, A., Kóczy, L., Gedeon, T., 2009. IFSA-EUSFLAT 2009 Robot Cooperation without Explicit Communication by Fuzzy Signatures and Decision Trees.

[4] Beckers, R., Holland, O., Deneubourg, J., 2000. Prerational Intelligence: Adaptive Behavior and Intelligent Systems Without Symbols and Logic, Volume 1, Volume 2 Prerational Intelligence: Interdisciplinary Perspectives on the Behavior of Natural and Artificial Systems, Volume 3. Studies in Cognitive Systems. Springer Netherlands, Dordrecht, Ch. From Local Actions to Global Tasks: Stigmergy and Collective Robotics, pp. 1008–1022.

[5] Beni, G., 2005. From swarm intelligence to swarm robotics. In: Swarm robotics. Springer, pp. 1–9.

[6] Biocyclopedia, 2007. Animal communication.
URL `http://www.eplantscience.com/index/general_zoology/animal_communication.php`

[7] Bishop, C. M., et al., 2006. Pattern recognition and machine learning. Vol. 4. Springer New York.

[8] Blum, R. S., Liu, Z., 2005. Multi-sensor image fusion and its applications. CRC press.

[9] Cao, Y., Fukunaga, A., Kahng, A., Meng, F., Aug 1995. Cooperative mobile robotics: antecedents and directions. In: Intelligent Robots and Systems 95. 'Human Robot Interaction and Cooperative Robots', Proceedings. 1995 IEEE/RSJ International Conference on. Vol. 1. pp. 226–234.

[10] Couceiro, M., Rocha, R., Vargas, P., June 2013. Darwinian Robotic Swarms for Exploration with Minimal Communication. In: 2013 IEEE Congress on Evolutionary Computation (CEC). pp. 127–134.

[11] De, C., Leandro, N., Von, Z., Fernando, J., 2005. Recent developments in biologically inspired computing. Igi Global.

[12] Deneubourg, J.and Goss, S., Sandini, G., Ferrari, F., Dario, P., 1990. Self-organizing collection and transport of objects in unpredictable environments. In: Symposium on Flexible Automation. pp. 1093–1098.

[13] Dorigo, M., Birattari, M., Brambilla, M., 2014. Swarm robotics. Scholarpedia 9 (1), 1463.

[14] Dornhaus, A., Chittka, L., 2003. Why do honey bees dance? Behavioral Ecology and Sociobiology 55 (4), 395–401.

[15] Esch, H., Burns, J., January 1996. Distance estimation by foraging honeybees. The Journal of experimental biology 199, 155–62.

[16] Ghosh, S., Marshall, I., 2005. Simple model of collective decision making during nectar source selection by honey bees. In: CD Rom of Workshop on Memory and Learning Mechanisms in Autonomous Robots.

[17] Gil, M., Marco, R., November 2010. Decoding information in the honeybee dance: revisiting the tactile hypothesis. Animal Behaviour 80 (5), 887–894.

[18] Gonzalez, R. C., Woods, R. E., 2006. Digital Image Processing. Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

[19] Hartley, R., Zisserman, A., 2003. Multiple view geometry in computer vision. Cambridge university press.

[20] Hernandez-Marin, S., Wallace, A., Gibson, G., Dec 2007. Bayesian analysis of lidar signals with multiple returns. Pattern Analysis and Machine Intelligence, IEEE Transactions on 29 (12).

[21] Higgins, F., Tomlinson, A., Martin, K. M., April 2009. Survey on security challenges for swarm robotics. In: Autonomic and Autonomous Systems, 2009. ICAS '09. Fifth International Conference on. pp. 307–312.

[22] Huber, M., Durfee, E., 1995. Deciding When to Commit To Action During Observation-based Coordination. In: In Proceeding of the First International Conference on Multi-Agent Systems (ICMAS-95). pp. 163–170.

[23] Janson, S., Middendorf, M., Beekman, M., December 2006. Searching for a new home–scouting behavior of honeybee swarms. Behavioral Ecology 18 (2), 384–392.

[24] Jennings, N., Jun 1995. Controlling cooperative problem solving in industrial multi-agent systems using joint intentions. Artificial Intelligence 75 (2), 195–240.
[25] Jiménez Lugo, J., Zell, A., October 2014. Framework for Autonomous On-board Navigation with the AR.Drone. Journal of Intelligent and Robotic Systems 73 (1-4), 401–412.
[26] KaewTraKulPong, P., Bowden, R., 2002. An Improved Adaptive Background Mixture Model for Real-time Tracking with Shadow Detection. In: Video-Based Surveillance Systems. Springer US, pp. 135–144.
[27] Kube, C., Zhang, H., 1993. Collective robotics : From social insects to robots. Adaptive behavior 2 (2), 189–218.
[28] Landgraf, T., Oertel, M., Kirbach, A., Menzel, R., Rojas, R., 2012. Imitation of the honeybee dance communication system by means of a biomimetic robot. In: Prescott, T., Lepora, N., Mura, A., Verschure, P. (Eds.), Biomimetic and Biohybrid Systems. Vol. 7375 of Lecture Notes in Computer Science.
[29] Landgraf, T., Rojas, R., Nguyen, H., Kriegel, F., Stettin, K., August 2011. Analysis of the waggle dance motion of honeybees for the design of a biomimetic honeybee robot. PloS one 6 (8), e21354.
[30] Lim, J. S., 1990. Two-dimensional signal and image processing. Englewood Cliffs, NJ, Prentice Hall, 710 p. Vol. 1.
[31] Melgani, F., Bruzzone, L., 2004. Classification of hyperspectral remote sensing images with support vector machines. IEEE Transactions on Geoscience and Remote Sensing 42 (8), 1778–1790.
[32] Montemerlo, M., Thrun, S., Koller, D., Wegbreit, B., et al., 2002. Fastslam: A factored solution to the simultaneous localization and mapping problem. In: AAAI/IAAI. pp. 593–598.
[33] Noble, J., Boukerroui, D., 2006. Ultrasound image segmentation: A survey. IEEE Transactions on Medical Imaging 25 (8), 987–1010.
[34] Novitzky, M., Pippin, C., Collins, T., Balch, T., West, M., December 2012. Bio-inspired multi-robot communication through behavior recognition. In: IEEE International Conference on Robotics and Biomimetics (ROBIO). pp. 771–776.
[35] Onn, S., Tennenholtz, M., 1997. Determination of social laws for multi-agent mobilization.
[36] Parker, L., April 1998. ALLIANCE: an architecture for fault tolerant multirobot cooperation. IEEE Transactions on Robotics and Automation 14 (2), 220–240.
[37] Parker, L., 2008. Handbook of Robotics. Springer, Ch. Chapter 40 : Multiple Mobile Robot Systems, pp. 921–941.
[38] Pham, D., Koc, E., Lee, J., Phrueksanant, J., 2007. Using the bees algorithm to schedule jobs for a machine. In: Proceedings of eighth international conference on laser metrology, CMM and machine tool performance. pp. 430–439.
[39] Piccardi, M., 2004. Background subtraction techniques: a review. In: Systems, man and cybernetics, 2004 IEEE international conference on. Vol. 4. IEEE, pp. 3099–3104.
[40] Rege, S., Memane, R., Phatak, M., Agarwal, P., 2013. 2D Geometric Shape And Color Recognition Using Digital Image Processing. International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering 2 (6), 2479–482.
[41] Rekleitis, I., Lee-Shue, V., New, A. P., Choset, H., 2004. Limited communication, multi-robot team based coverage. In: Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on. Vol. 4. IEEE, pp. 3462–3468.
[42] Russell, S., Norvig, P., December 2009. Artificial Intelligence: A Modern Approach, 3rd Edition. Prentice Hall.
[43] Schultz, A. C., Parker, L. E., 2002. Multi-robot systems: from swarms to intelligent automata. Springer.
[44] Seeley, T., Mikheyev, A., Pagano, G., October 2000. Dancing bees tune both duration and rate of waggle-run production in relation to nectar-source profitability. Journal of Comparative Physiology A: Sensory, Neural, and Behavioral Physiology 186 (9), 813–819.
[45] Sharp, C. S., Shakernia, O., Sastry, S. S., 2001. A vision system for landing an unmanned aerial vehicle. In: IEEE International Conference on Robotics and Automation, 2001. Proceedings 2001 ICRA. Vol. 2. pp. 1720–1727.
[46] Shim, J., Arkin, R. C., 2014. Robot deception and squirrel behavior: A case study in bio-inspired robotics. Tech. rep., DTIC Document.

[47] Soille, P., 2003. Morphological image analysis: principles and applications. Springer-Verlag, New York, Inc.
[48] Stauffer, C., Grimson, W. E. L., August, 1999. Adaptive background mixture models for real-time tracking. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition,. Vol. 2. pp. 2246–252.
[49] Szeliski, R., 2010. Computer vision: algorithms and applications. Springer Science & Business Media.
[50] Szeliski, R., 2010. Computer Vision: Algorithms and Applications, 1st Edition. Springer-Verlag New York, Inc., New York, NY, USA.
[51] Tambe, M., 1997. Towards Flexible Teamwork. Journal of Artificial Intelligence Research 7, 83–124.
[52] Vargas, P. A., Di, P., Ezequiel, A., Harvey, I., Husbands, P., 2014. The horizons of evolutionary robotics. MIT Press.
[53] Von Frisch, K., 1967. The dance language and orientation of bees. Harvard University Press.
[54] Yan, Z., Jouandeau, N., Cherif, A. A., 2013. A survey and analysis of multi-robot coordination. International Journal of Advanced Robotic Systems 10.